Matrix Multiplication

On CCR Cluster

CSE 633 Parallel Algorithms
Spring 2014
Praveen Kumar Bellam
Outline

• Sequential Algorithm
• Parallel Algorithm
• Parallel Implementation Using MPI
• Parallel Implementation Using Open MP
• Results
**Input:** Two square matrices A, B of size $n \times n$  
**Output:** The matrix product $C_{n \times n} = A \times B$

**Matrix Multiplication (A, B)**

for $i = 1$ to $n$ do  
    for $j = 1$ to $n$ do  
        $c_{ij} = 0$  
        for $k = 1$ to $n$ do  
            $c_{ij} = c_{ij} + a_{ik}b_{kj}$  
        end for  
    end for  
end for  

$\Theta(n^3)$
Parallel Algorithm – Design Considerations

• Distributing the data
• Local computation
• Communicating the data (Send/Receiving)
• Gathering the results
Parallel Algorithm

1. Divide the input matrices into P blocks. P is the number of processors available for computation.
2. Create a matrix of processes of size $P^{1/2} \times P^{1/2}$ so that each process can maintain a block of A matrix and a block B matrix.
3. Each block is sent to each process by determining the owner, and the copied sub blocks are multiplied together and the results added to the partial results in the C sub-blocks.
4. The A sub-blocks are rolled one step to the left and the B sub-blocks are rolled one step upwards.
5. Repeat the process, $P^{1/2}$ times.
Divide input matrices into P sub blocks, and distribute the data

Input Matrix $A_{4 \times 4}$

\[
\begin{bmatrix}
  a_1 & a_2 \\
  a_5 & a_6 \\
  a_9 & a_{10} \\
  a_{13} & a_{14}
\end{bmatrix}
\quad \begin{bmatrix}
  a_3 & a_4 \\
  a_7 & a_8 \\
  a_{11} & a_{12} \\
  a_{15} & a_{16}
\end{bmatrix}
\]

Input Matrix $B_{4 \times 4}$

\[
\begin{bmatrix}
  b_1 & b_2 \\
  b_5 & b_6 \\
  b_9 & b_{10} \\
  b_{13} & b_{14}
\end{bmatrix}
\quad \begin{bmatrix}
  b_3 & b_4 \\
  b_7 & b_8 \\
  b_{11} & b_{12} \\
  b_{15} & b_{16}
\end{bmatrix}
\]
Each processor performs local matrix multiplication of blocks.
Each processor, sends A’s sub block to the processor on the left, B’s sub block to the processor above.
Again, perform local matrix multiplication and add it to the result set.
<table>
<thead>
<tr>
<th>Number of Processors</th>
<th>2000</th>
<th>3000</th>
<th>4000</th>
<th>5000</th>
<th>6000</th>
<th>7000</th>
<th>8000</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>24</td>
<td>85</td>
<td>280</td>
<td>464</td>
<td>1125</td>
<td>1732</td>
<td>2443</td>
</tr>
<tr>
<td>4</td>
<td>9.43</td>
<td>23.70</td>
<td>70.90</td>
<td>256.62</td>
<td>588.38</td>
<td>988.20</td>
<td>1297.69</td>
</tr>
<tr>
<td>9</td>
<td>6.85</td>
<td>20.50</td>
<td>42.56</td>
<td>144.59</td>
<td>390.40</td>
<td>488.72</td>
<td>620.96</td>
</tr>
<tr>
<td>16</td>
<td>5.20</td>
<td>15.10</td>
<td>35.16</td>
<td>97.14</td>
<td>281.76</td>
<td>359.46</td>
<td>257.72</td>
</tr>
<tr>
<td>25</td>
<td>1.33</td>
<td>6.80</td>
<td>14.79</td>
<td>67.08</td>
<td>107.91</td>
<td>235.77</td>
<td>186.07</td>
</tr>
<tr>
<td>36</td>
<td>4.20</td>
<td>12.70</td>
<td>24.01</td>
<td>146.51</td>
<td>187.40</td>
<td>380.78</td>
<td>580.31</td>
</tr>
<tr>
<td>49</td>
<td>5.72</td>
<td>13.78</td>
<td>36.78</td>
<td>153.08</td>
<td>192.66</td>
<td>413.12</td>
<td>603.64</td>
</tr>
<tr>
<td>64</td>
<td>6.80</td>
<td>18.30</td>
<td>44.43</td>
<td>158.65</td>
<td>202.74</td>
<td>419.35</td>
<td>613.37</td>
</tr>
<tr>
<td>NUMBER OF PROCESSORS</td>
<td>Matrix Size</td>
<td>2000</td>
<td>3000</td>
<td>4000</td>
<td>5000</td>
<td>6000</td>
<td></td>
</tr>
<tr>
<td>----------------------</td>
<td>-------------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>24</td>
<td>85</td>
<td>280</td>
<td>464</td>
<td>1125</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>11.36</td>
<td>44.35</td>
<td>132.12</td>
<td>267.05</td>
<td>403.06</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>8.23</td>
<td>31.66</td>
<td>99.08</td>
<td>206.26</td>
<td>362.38</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>6.57</td>
<td>23.43</td>
<td>71.32</td>
<td>139.01</td>
<td>253.85</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>4.44</td>
<td>15.66</td>
<td>57.31</td>
<td>115.05</td>
<td>204.02</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>3.26</td>
<td>9.30</td>
<td>45.54</td>
<td>88.13</td>
<td>162.54</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>2.44</td>
<td>7.30</td>
<td>37.75</td>
<td>86.69</td>
<td>127.68</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>2.37</td>
<td>6.02</td>
<td>25.47</td>
<td>52.52</td>
<td>109.34</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Observations & Learnings

- Increasing the processors doesn’t always reduce the running time.
- Need to experimentally identify the point where communication costs are taking over the local computation.
- Running times depend on how the nodes got allocated on CCR cluster.
- Need to specify in the SLURM Script about the node details.
- Got good understanding about parallelization.
- Next step is to analyze and understand the semantics of parallel architectures practically by simulating the algorithms.
References